WHAT IS CLAIMED IS:

1. A system for shortening a class loading process in a Java program, comprising:

a class loader unit for loading Java program class files from an auxiliary memory, performing linking and initialization processes and generating runtime data;

a first memory unit for maintaining the runtime data generated by the class loader unit in an accessible state;

a second memory unit for storing the runtime data, which have been loaded into the first memory unit in the accessible state, in a form of images;

a runtime data search unit for loading the runtime data, which have been stored in the second memory unit in the form of images, into the first memory unit upon the request of the class loader unit; and

an execution unit for executing the runtime data that have been loaded into the first memory unit in the accessible state.

2. The system as claimed in claim 1, further comprising a garbage collector unit for collecting space unused in the first memory unit and allowing the unused space to be used again.

3. The system as claimed in claim 1, wherein the runtime data

search unit causes the runtime data generated by the class loader unit to be stored in the second memory unit in the form of images.

4. The system as claimed in claim 1 , wherein the runtime data search unit manages the runtime data, which have been stored in the second memory unit in the form of images, by using a least recently used (LRU) method.

5. The system as claimed in claim 3, wherein the runtime data search unit manages the runtime data, which have been stored in the second memory unit in the form of images, by using a least recently used (LRU) method.

6. A method for shortening a class loading process in a Java program, comprising the steps of:

requesting a runtime data search unit to search runtime data necessary for execution of the Java program, by a class loader unit;

searching the requested runtime data for the Java program by the runtime data search unit;

transmitting the searched runtime data to a first memory unit; and

executing the runtime data transmitted to the first memory unit.

7. The method as claimed in claim 6, wherein the searched runtime data are stored in a second memory unit in a form of images.

8. The method as claimed in claim 7, wherein the runtime image data stored in the second memory unit are managed by the runtime data search unit according to a least recently used (LRU) method.

9. The method as claimed in claim 6, further comprising the steps of, if it is determined from search results of the requested runtime data for the Java program that there are no relevant runtime data,

loading Java program class files from an auxiliary memory;

generating runtime data by performing linking and initialization processes of the loaded Java program class files;

storing the generated runtime data in a form of images; and

transmitting the runtime image data to the first memory unit.

10. The method as claimed in claim 9, wherein the step of storing the generated runtime data in the form of images is performed after the step of executing the runtime data transmitted to the first memory unit.

11. The method as claimed in claim 9, wherein the stored runtime image data are managed by the runtime data search unit according to a least recently used (LRU) method.